ORACLE
INSURANCE

Oracle Health Insurance Back Office

Service Callout Installation & Configuration Manual

Version 1.0

Part number: G24367-01

February 20, 2025

CHANGE HISTORY

Release	Version	Changes
10.25.2.0.0	1.0	• Creation
		•

RELATED DOCUMENTS

A reference in the text (**doc[x]**) is a reference to another document about a subject that is related to this document.
Below is a list of related documents:

Doc[1]	OHI Back Office - Installation & Configuration Manual (docs.oracle.com)
Doc[2]	OHI Back Office - Service Consumer Installation & Configuration Manual (docs.oracle.com)
Doc[3]	APEX Install and Upgrade guide
Doc[4]	Oracle 19c Database Security Guide
Doc[5]	APEX API Reference

Contents

1 Introduction

The OHI Back Office web service consumers are designed to call third-party web services or web services that have an interface that has been defined by OHI. These web services are present in the world outside OHI Back Office and are called from the PL/SQL code within the OHI Back Office database, with several intermediary steps.

Each web service consumer has a corresponding PL/SQL wrapper package providing a PL/SQL interface to the public methods of the web service consumer.

The PL/SQL wrapper functions are called from within the OHI Back Office database and make use of Oracle Application Express (APEX) PL/SQL API packages.

1.1 Licenses

No additional OHI Back Office license option is required to use the web service consumers. The web services (incoming service calls) and web service consumers (outgoing service calls) together constitute the 'Service Layer'. The web services part of the 'Service Layer' DO require an additional OHI license.

For further information, please consult your OHI sales representative.

For the underlying technology, Oracle Database, you have to acquire separate technology licenses; these are not included within OHI Back Office.
APEX is available as a free, no-cost feature of the Oracle Database. The entire stack, including Oracle APEX and Oracle REST Data Services (ORDS) are fully supported with your Oracle Database license.

2 Architectural overview

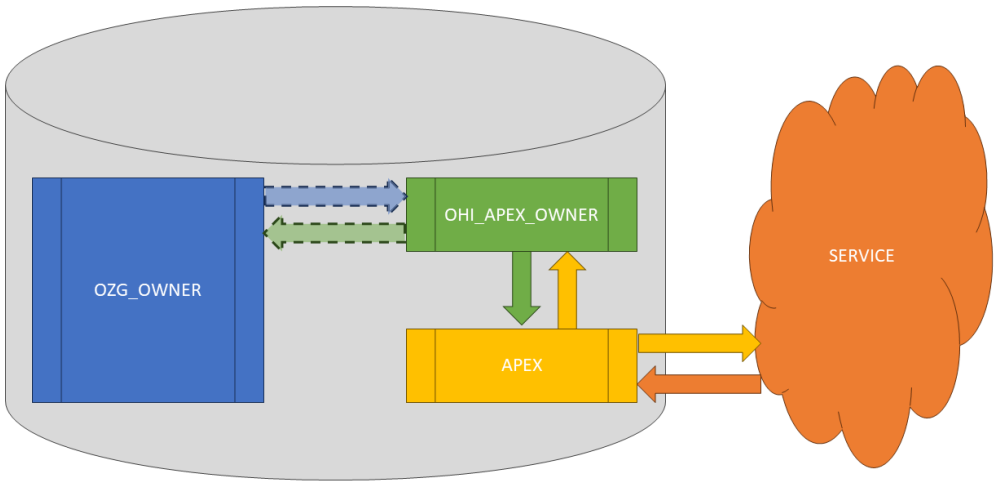
The service consumer functionality as described below offers synchronous calls to external web services. This means OHI calls to the web service are ended when the call returns a response or a time-out is reached.

This chapter gives a high level architectural overview of the web service consumers, in short 'SVC'. The abbreviation SVC is used in many OHI module names to make clear the files belong to the **S**er**v**ice **C**onsumer implementation, which makes use of APEX PL/SQL API packages.

The implementation of the SOAP service consumers using the Oracle Service Bus (OSB) and JMS-queues is described in the *Oracle Health Insurance Back Office - Service Consumer Installation and Configuration Manual*.

2.1 Design

The diagram below shows how the OHI Back Office schema owner interacts with an external service operation



The solution is based on the following architecture:

- The OHI Back Office application calls the OHI interface package with the request from within a PL/SQL routine in the OHI Back Office owner schema.
- The interface package in the OHI_APEX_OWNER schema calls the APEX API to perform the actual service callout.
- The OHI_APEX_OWNER schema resides in the same pluggable database as the APEX installation.
- The communication between the schema owner of OHI Back Office (usually OZG_OWNER) and the OHI_APEX_OWNER schema is always via a database link, even if both schemas are in the same pluggable database.
- The response status and message are propagated back to the calling function in the OHI Back Office schema.

- A timeout will occur if the response from the service callout is not returned within the given timeout period.

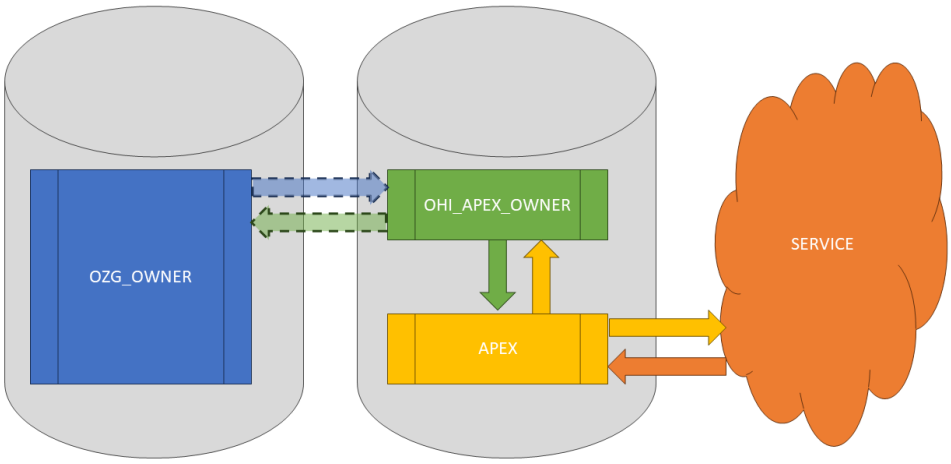
2.2 Alternative configurations

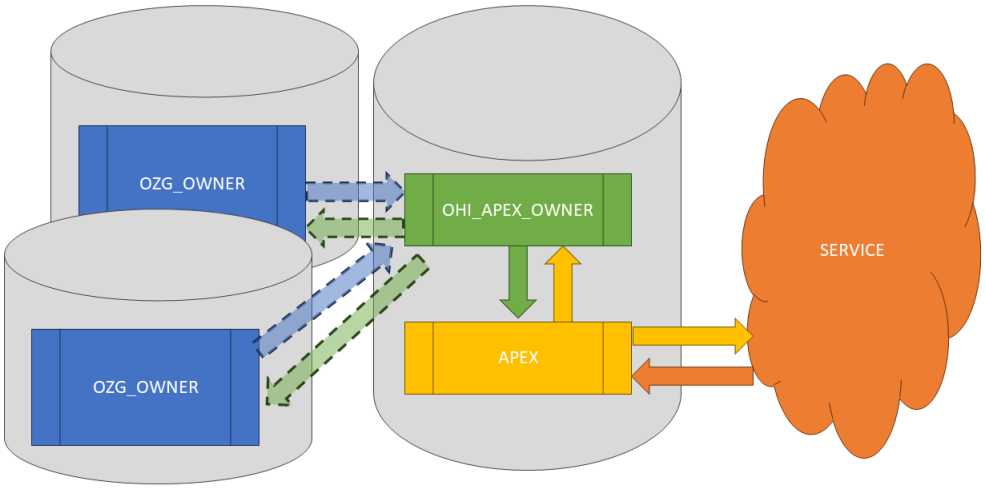
Besides the configuration where the APEX schema is in the same pluggable database a configuration with APEX in a separate pluggable database is also an option. OHI prefers the option where the APEX schema is in the same pluggable database as the OHI Back Office application.
Future implementations may no longer support the configuration with APEX in a separate pluggable database.

A separate pluggable database with an APEX installation may be required if you currently use APEX in the pluggable database of OHI Back Office and an upgrade is not (yet) possible.

A separate pluggable database with an APEX installation can also be used to service multiple non-production environments and so limit the amount of configuration required.

The diagrams below show the different configurations, where on or more OHI Back Office environments are connected to one APEX environment. The first one is a typical production configuration and the second one is a possible configuration for test environments.
The configuration of the endpoints (URLs) is done in OHI Back Office. This way one OHI Back Office environment can call a test endpoint and another one can call the acceptance endpoint of the same service using the same APEX installation.




OZG_OWNER
OHI_APEX_OWNER
SERVICE
OZG_OWNER
APEX

3 Prerequisites

The functionality of the service consumers may not be required in all OHI BO environments. It is certainly needed in the Production environment and in some integration test environments. In environments where the service consumers are needed, this functionality must be enabled by fulfilling the prerequisites described below.

The deployment itself is described in the next chapter.

3.1 APEX

Oracle Application Express (APEX) should be installed in a pluggable database. This can be the same pluggable database (PDB) where the OHI Back Office application is installed, or a separate pluggable database, either in the same or a different Container Database (CDB).
For release 10.25.2.0.0 the required APEX version is 24.1. OHI may require higher versions of APEX with different OHI releases or interim patches.

More on the recommendation to install APEX in a pluggable database and alternatives can be found in chapter 5 (Utilizing Multitenant Architecture) of the *APEX install and Upgrade* guide.

As OHI Back Office solely uses the APEX PL/SQL APIs only the installation of APEX described in chapter 6.3.1 of the *APEX Install and Upgrade* guide is required. The creation of the instance administration account, public user and router account can be omitted.

Note: Currently the installation of the Oracle REST Data Services (ORDS) is not required.

For more information, please use the database documentation and consult the *APEX Install and Upgrade* guide.

3.2 OHI_APEX_OWNER

In the same pluggable database where APEX is installed, a database account OHI_APEX_OWNER must be created/present. This account will handle the communication between the OHI Back Office and the APEX schemas.
The required grants and settings for this database account are described in the next chapter.

The required software for this database account to handle the communication will be installed via OHIPATCH.

3.3 Database link

Create a private database link from the OHI Back Office schema owner to the OHI_APEX_OWNER. Register the name of this database link in the OHI Back Office parameter value for parameter 'Database link' in the Back Office parameter group 'Service Callouts'. This parameter will be used by OHIPATCH to determine the correct database link and perform the installation of the applicable sources in the OHI_APEX_OWNER schema. OHIPATCH does not require direct access to the OHI_APEX_OWNER schema.

4 Deployment Instructions

This chapter explains

- how to configure OHI_APEX_OWNER
- how to configure the Access Control List (ACL)
- how to store credentials

4.1 Configure OHI_APEX_OWNER

In the same pluggable database where APEX is installed a dedicated database account is required for use with OHI Back Office.

The database account OHI_APEX_OWNER needs the following system privileges. Grant these via a user with DBA privileges.

- grant create session to ohi_apex_owner;
- grant alter session to ohi_apex_owner;
- grant select any dictionary to ohi_apex_owner;
- grant create procedure to ohi_apex_owner;
- grant execute on DBMS_NETWORK_ACL_ADMIN to ohi_apex_owner;

Some configuration in the APEX schema is also required for OHI_APEX_OWNER to be able to perform the callouts via the APEX APIs.

```
begin
  apex_instance_admin.add_workspace
  ( p_workspace      => 'OHIBO_WORKSPACE'
  , p_primary_schema => 'OHI_APEX_OWNER'
  );

  apex_instance_admin.set_workspace_parameter
  ( p_workspace => 'OHIBO_WORKSPACE'
  , p_parameter => 'MAX_WEBSERVICE_REQUESTS'
  , p_value     => 10000000
  );
end;
/
```

These statements are also available in the script OZGI008S.sql. That script is part of the 10.25.2.0.0 release and will be installed in the /sql directory of the OHI Back Office application.

4.2 Configure the Access Control List

To be able to call a service from inside the database, a schema or user must be granted access to the service location. This is done via the access control list (ACL).

The configuration of the access control list is done via the database package DBMS_NETWORK_ACL_ADMIN.

See the documentation for this package in the *PL/SQL Packages and Types Reference* for further information.

The DBMS_NETWORK_ACL_ADMIN package is available via the OHI_APEX_OWNER account.
The 'principal' for the so called ACE should be the owner account of the current APEX schema, eg. APEX_240100 for APEX version 24.1.

The statement to allow a http(s) connection to, for instance , host myhost.company.com on port 6512 can look like this:

```
declare
  l_principal varchar2(20) := 'APEX_240100';
begin
  dbms_network_acl_admin.append_host_ace
  ( host       => 'myhost.company.com'
  , lower_port => 6512
  , upper_port => 6513
  , ace        => xs$ace_type
                  ( privilege_list => xs$name_list
                                      ( 'http'
                                      , 'https'
                                      )
                  , principal_name => l_principal
                  , principal_type => xs_acl.ptype_db
                  )
  );
end;
/
```

When SSL-certificates are needed to perform the callout, a wallet containing these certificates should be accessible from the database instance.
This can also be configured using the DBMS_NETWORK_ACL_ADMIN package:

```
declare
  l_principal varchar2(20) := 'APEX_240100';
begin
  dbms_network_acl_admin.append_wallet_ace
  ( wallet_path => 'file:///ohi/keyBase/wallets/httpscerts'
  , ace         => xs$ace_type
                   ( privilege_list => xs$name_list
                                       ( 'use_client_certificates'
                                       , 'use_passwords'
                                       )
                   , principal_name => l_principal
                   , principal_type => xs_acl.ptype_db
                   )
  );
end;
```

How to create such wallet (httpscerts in this example) with orapki is described in the *Oracle 19c Database Security Guide* appendix F "Managing Public Key Infrastructure (PKI) Elements".

The setup can be checked via the following database views:
select * from DBA_HOST_ACES where principal = [APEX OWNER];

select * from DBA_WALLET_ACES where principal = [APEX OWNER];

4.3 Secure storing of credentials

The endpoint of the service operation can require a form of authentication like Basic Authentication or OAUTH2.
The credentials for this endpoint can be stored in a credential store in the APEX database. The objectname of the credential store can be used by the application to pass these credentials with the callout to the service operation, hiding the actual credentials from the OHI Back Office application.

An example to configure a basis authentication credential:

```
   begin
     apex_util.set_workspace (p_workspace =>'OHIBO_WORKSPACE');
     apex_credential.create_credential
     ( p_credential_name      => 'OHIBO_CREDENTIALS'
     , p_credential_static_id => 'OHIBO_CREDENTIALS'
     , p_authentication_type  => apex_credential.C_TYPE_BASIC
     , p_allowed_urls         => apex_t_varchar2
                           ('https://myhost.company.com'
                           ,'http://myhost.company.com')
     , p_prompt_on_install    => true
     , p_credential_comment   => 'OHIBO basic store'
     );

  apex_credential.set_persistent_credentials
  ( p_credential_static_id => 'OHIBO_CREDENTIALS'
  , p_username             => 'my_user'
  , p_password             => 'mySecretPassword123'
  );
end;
/
```

NB. Note the use of the correct workspace (OHIBO_WORKSPACE) that was created above.

Access to a wallet containing https certificates can be stored in a credential store. This credential store can/will not be passed in each call from OHI Back Office but is set at system level in the APEX schema. The access to this wallet is described in the previous paragraph.

An example of configuring the use for such wallet credential store for a wallet with the name httpscerts can be as follow:

```
begin
   apex_instance_admin.set_parameter
  ( p_parameter => 'WALLET_PATH'
  , p_value     => 'file:///ohi/keyBase/wallets/httpscerts'
  );

  apex_instance_admin.set_parameter
  ( p_parameter => 'WALLET_PWD'
  , p_value     => 'mySecretPassword123'
  );
end;
/
```

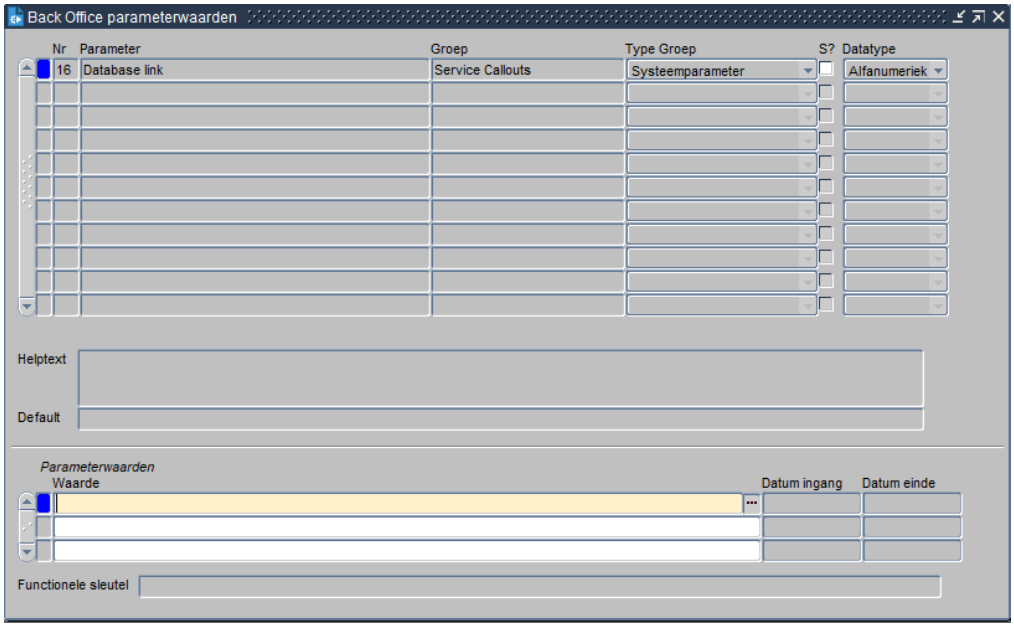
See the *Oracle APEX API Reference* for more information about configuring and maintaining the credential store(s).

5 Back Office configuration

Part of the configuration is done in the OHI Back Office application.

5.1 General setting

The name of the database link from the OHI Back Office schema to the OHI_APEX_OWNER schema needs to be entered in the OHI Back Office parameter "Database link" in the system parameter group "Service Callouts".



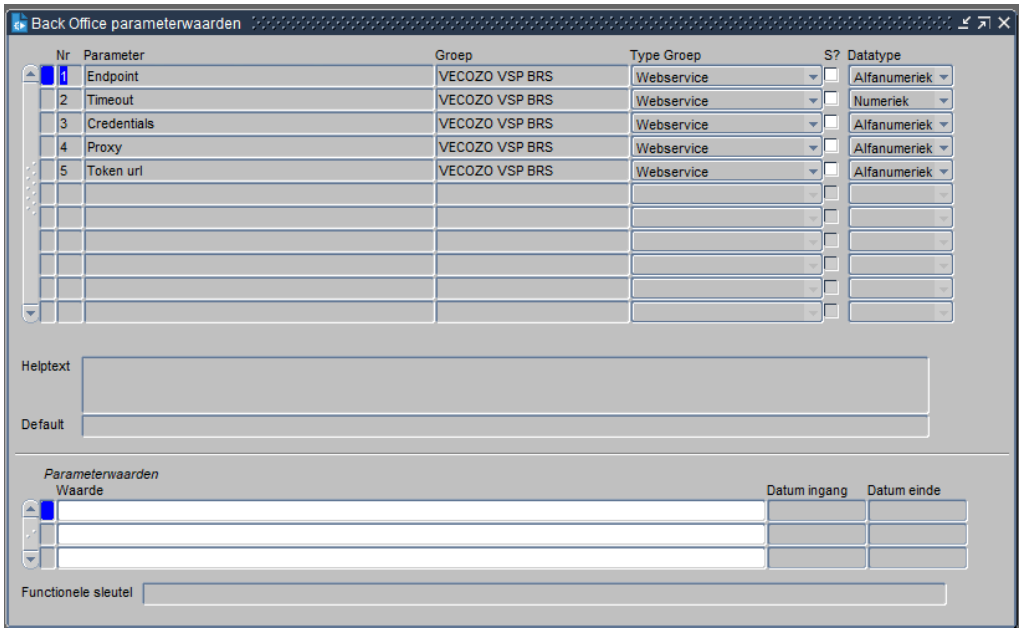
The registered database link name will be used in OHIPATCH in step 110 to install and configure the interface between the OHI Back Office schema and the OHI_APEX_OWNER schema.

If it is not possible to switch from database link during a release installation, the installation and/or configuration can also be executed via sqlplus under the OHI Back Office schema account or via a DBA-role.

```
begin
  sys_install_svc_pck.run;
end;
/
```

5.2 Service specific settings

The configuration for the services can be set in the Back Office parameter values screen (SYS1145F) of the application instance. All parameters can be found in an own group for each individual web service that can be consumed, e.g. "VECOZO VSP BRS" for the VECOZO VPS service in REST:



Each service has five parameters

- Endpoint: The URL endpoint of the web service
- Timeout: The amount of time in seconds to wait for a response.
- Credentials: The name of the credential to be used (see paragraph 4.2).
- Proxy: The proxy server to use for the request
- Token url: For token-based authentication flows (like OAuth2): The URL where to get the token from.

The endpoint should include the path parameters like the parameter contractId in the next sample: "/polis/v1/collectievecontracten/{contractId}/polissen".
Query parameters should be omitted from the endpoint. During the actual call the path parameters will be replaced with the correct value and the query parameters will be added to the UR, with the correct values if applicable.

An alternative to a database wallet to store HTTPS certificates is to use a proxy server. With this approach, you call an HTTP end-point, and the forward proxy passes the request on to the final end-point over HTTPS.

The value set for the timeout parameter is a maximum. Setting a high value could cause a screen or batch session seem to "hang" waiting for the response during at the most the specified nr of seconds.

Note: The value of a Back Office parameter is cached when called for the first time. Services handled by the autonomous event framework, like the VECOZO services, may use the cached value for up to an hour.
It is advised to stop and start the batch scheduler after changing the time-out to reinitialize this cached value.

6 Monitoring and Troubleshooting

This chapter contains some troubleshooting paragraphs. These can be used if the service consumer calls do not work and result in errors that are not functional by nature.

6.1 Monitoring

All calls are logged in the table ALG#SERVICE_CONSUMER_LOG

Column	Description
ID	Unique identification for each row
SERVICENAAM	The functional (OHI) name of the service
BERICHT_TYPE	Functional message code; for the VECOZO service this is the actual attribute 'bericht type'
BERICHT_SUBTYPE	For the VECOZO service this is the actual attribute 'bericht subtype'
BERICHT_ACTIE	For the VECOZO service this is the actual attribute 'bericht actie'
STARTTIJD_AANVRAAG	Time the call to the endpoint was made
EINDTIJD_AANVRAAG	Time a response (or timeout) was received from the endpoint or (APEX)API
TYPE_OPERATIE	Type of operation, e.g. GET or POST
RESPONSE_STATUS	The response (http) status of the request
LENGTE_AANVRAAG	Size in bytes of the request
LENGTE_ANTWOORD	Size in bytes of the response
AANVRAAG	The endpoint (url) used in the callout including the resource and query parameters
ERROR_TEKST	The errormessage, if an error occured during the callout

The same information is available in the batch report "Overzicht autonome verwerking (SYS5002R)" and in the dashboard of the Autonous Processing Framework (AVF) in the OHIJET application. Look for the Services Consumers sections.